3D Simulator of Modular Building Assembly by Automated Cranes

R. Diez, V. M. Padrón, M. Abderrahim, C. Balaguer

University Carlos III of Madrid, Department of System Engineering and Automation
C/Butarque, 15, 28911 Leganés, Madrid (Spain)
e-mail: rdiez@ing.uc3m.es

ABSTRACT: This paper presents a new construction method based of modular buildings. In this new method, the buildings are to be assembled from their modules using automatic means such as robots or/and automated cranes. A Computer Integrated Construction (CIC) architecture has been proposed to achieve the modular construction covering all aspects from the design until the assembly on site. The system is modular: design, planning and simulation tools are integrated in a graphical user interface. The 3D building design is performed by special tools developed for this purpose. Then planning tools calculate automatically the modules assembly sequence and the path to place the modules by the crane. These trajectories are animated with the developed simulation tools to ensure secure paths and to correct the erroneous ones. Finally, an automatic gantry crane developed at the Laboratory is used to test the automatic modules placement.
This paper is dedicated to describe the simulation tools. Two crane simulators for a gantry and a tower crane, a specific programming language and a program editor have been developed. A program contains a list of instructions with the trajectories that can be automatically generated by a planning tool or manually by the operator using the program editor integrated in the simulator. The user can select different velocities to run a program and can execute a program instruction by instruction or in continuous mode. A toolbar shows the position of the grasping platform at each moment and the current instruction in execution. The user can also move manually the simulator by a graphical pendant that has buttons to move each joint individually. This program can be used in the laboratory gantry crane to place the modules to assemble a particular building setup.

KEYWORDS: 3D animation, crane simulation, robot programming language, modular building, CIC.

1. INTRODUCTION

Traditional methods of house-building are usually based on manual techniques which are slow and expensive. New construction methods based on modular buildings will not only increase productivity, but also improve work safety and reduce the numerous hazards of nowadays work in building sites. The modules are prefabricated in a well controlled environment and the transported to the site. In this new method, the buildings are to be assembled from their modules using automatic means such as robots or/and automated cranes [Balaguer].
A Computer Integrated Construction (CIC) architecture has been proposed to achieve the modular construction covering all aspects from the design until the assembly on site. The system is modular, object-oriented, flexible and easy to extend. A graphical user interface integrates design, planning and simulation tools. The process starts with the 3D building design performed by special tools developed for this purpose. Then several planning tools are used to calculate automatically the modules assembly sequence. These modules will be placed on site by an automated crane. According to the established sequence the path to place the modules by the crane are also calculated automatically. These trajectories are animated with the developed simulation tools to ensure secure paths and to correct the erroneous ones. Finally, the automatic gantry crane developed at the Laboratory is used to test the automatic modules placement.
This paper is dedicated to describe the simulation tools. Two crane simulators for a gantry and a tower crane have been developed. A specific programming language has been created to write the task, which has to be executed by the crane. A program contains a list of instructions with the trajectories that can be automatically generated by a planning tool or manually by the operator using a text editor or the program editor integrated in the simulator. The simulator executes the program to visualise the assembly sequence. The user can select different velocities to run a program and can execute a program instruction by instruction or in continuous mode. A graphical pendant shows the

position of the grasping platform at each moment and the instruction in execution. The user can also move manually the simulator by a dialog box that has buttons to move each joint individually. The editor permits to write or modify a program and make sure that the program is error free. This program can be used in the laboratory gantry crane to place the modules to assemble a particular building setup.

2. SYSTEM ARCHITECTURE

The modular construction system is formed by a set of applications or modules and a group of data libraries, which are integrated in a Computer Integrated Construction (CIC) scheme [Diez 2003]. Applications or tools are grouped by their function in design, planning and simulation tools (figure 1 shows a schematic representation of the software architecture). The system is modular, object-oriented and flexible. New tools can been added to improve the system performance and to extend its capacities.

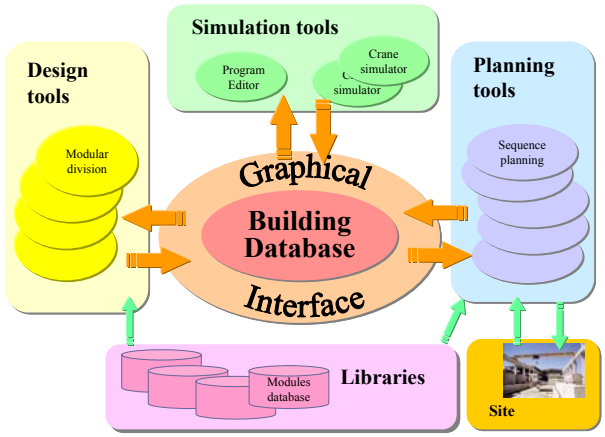

Figure 1. System architecture.

AutoCAD has been selected as the main software package, because of its widespread use and because it incorporates several development tools. Different technologies are available in AutoCAD to design and implement new AutoCAD-based applications [McAuley]. The chosen development tool for the implementation is the AutoCAD Runtime Extension (ObjectARX) [Owen Ransen]. ObjectARX is an Application Programming Interface (API) that contains a set of C++ classes and functions. A group of new functions have been created in order to enlarge and adapt AutoCAD to the problem specification. To facilitate the use, these additional software utilities are included in the AutoCAD menu bar.

A central database contains the information that can be accessed by applications through a graphical user interface. The AutoCAD *DWG* format is the file that store graphical data and other non-graphical information. AutoCAD is used as the main graphical user interface. Applications are created using ObjectARX and integrated into AutoCAD.

The designer produces the 3D building design by using special tools developed for this purpose. Modules, that form the spaces of a building, are included in the drawing by the designer. Dialog boxes guide the user to select the modules specifications contained in a library. Another tool, automatically, calculates the number, dimensions and characteristics of the modules needed to erect a building from the 2D building drawings [Diez 2000]. These data are saved into the building database.

Planning tools read data produced in the design stage. Then, these data are analysed in order to calculate production schemes, modules assembly sequence, transportation, etc. A planning applications determine the order in which modules are assembled and calculate automatically the trajectories that automated devices, such a crane, has to follow to place modules in their final position [Padron].

These paths, calculated automatically with planning tools, can be tested in a simulator before executing the commands in the real crane. The simulator is integrated with the rest of applications in the same graphical interface. Crane simulators, program editor and programming language are described in the following sections.

The designed and prefabricated modules are expected to be placed on site by an automated crane. An automatic gantry crane developed at the Laboratory is used to test the automatic modules placement [Abderrahim]. During the assembly process, the crane follows the paths previously calculated in the planning tool. The commands sequence is written in the same language used in the simulator.

The system includes a group of libraries, which contain information and data of the system: the module characteristics, the construction criteria, the modular division specifications' information, the structural specifications, etc.

3. SIMULATION TOOLS

Simulation tools are applications written in C++ object oriented language using ObjectARX and MFC (Microsoft Foundation Classes) libraries. They are compiled as DLLs (Dynamic Link

Library) that can be launch into AutoCAD at any moment.
Tower and gantry crane simulators, a programming editor and a programming language are the tools developed to simulate the modules automatic assembly.

3.1 Crane models

Two cranes models have been implemented: a gantry and a tower crane. Nevertheless, new types of cranes can be added because the system is modular. The cranes are drawn with prismatic models to represent the main structures and a line to simulate the elevation cable (see figures 2 and 3). These models are parametric, where the main crane dimensions (length of bridge or the jib, crane height, etc.) can be selected before drawing the model.
Crane dimensions are internal data to determine the crane reach. Position and orientation of the grasping tool determines the configuration of the crane. Current joint value is calculated via inverse kinematics. Graphical elements are displayed in their correct position and orientation calculating the homogeneous matrix corresponding to these values. Joints values are calculated in intervals to visualize a continuous movement during the simulation.

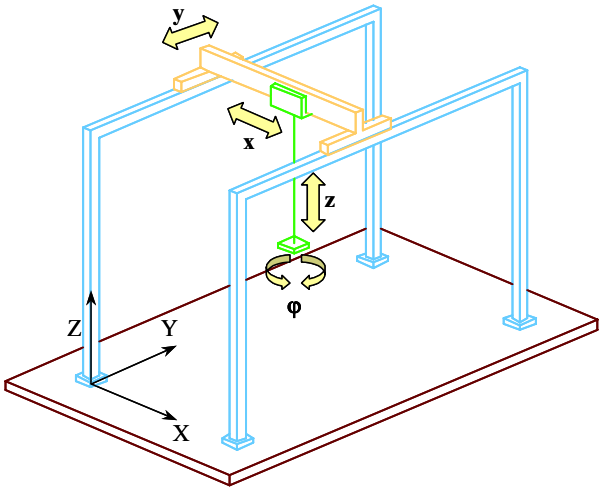

Figure 2. Model of a gantry crane

Gantry crane have tree movements implemented across Cartesian axes. It is possible to incorporate a fourth movement, a rotation of the grasping tool, to orientate the modules (see figure 2). These movements are:

- In the **x** axis, a linear movement of the trolley along the bridge.
- In the **y** axis the translation of the bridge.
- In the **z** axis, the vertical movement to rise/descent the module.
- A rotation along the **z** axis to orientate the module in xy plan.

Tower crane is drawn with the union of triangular and rectangular prisms to form the tower and the jib, lines to draw the cables that support the jib and counter-jib and a line with variable length to represent the hoisting cable (see figure 3).
Four movements have been implemented to simulate tower cranes:

- Slewing of the jib over the tower.
- Trolleying over the rails.
- Hoisting of the load.
- Rotation of the load along a vertical axis to orientate the modules being handled.

The joints can be moved separately and imitate the movement of a real tower crane using a joystick. It is also possible to perform other movements, such as linear movements, using a programming language. In these cases, two or more joints are moved simultaneously.

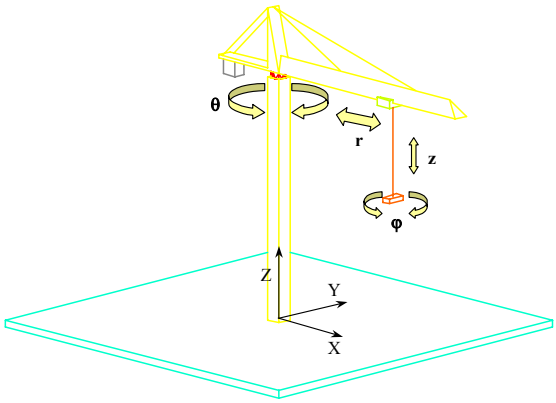

Figure 3. Model of a tower crane

3.2 Programming language

There is no universal programming language of robots, each manufacturer develop one specific for their products. A robot programming language should be easy to learn, similar to computer programming languages and does not change over time [Lapham].
The movements of a gantry crane (Cartesian robot) or a tower crane (cylindrical robot) are well known. When a module is placed in its final position, it is part of the environment. The paths are different for each module. According to these reasons, the language needed to program cranes should be simple and adaptable to different types of cranes.

The developed robot programming language has a simple syntax. A program is a text file containing a list of sentences. Each line contains a command followed by parameters if it is required; each command ends in a semicolon:

```
command_name [(par1, par2, par3);]
```

Language commands can be classified in different types:

- Absolute movement to destination: movement to point indicated in the parameters (*MoveTo, MoveJoints*) or movement in a Cartesian axis to the coordinate indicated in the parameter (*MoveToX, MoveToY, MoveToZ*).
- Relative movement: movement in a Cartesian axis from the current position to the distance indicated in the parameter (*MoveDistX, MoveDistY, MoveDistZ*).
- Joint movement: movement of each joint individually. The movement can be absolute (*RotateJibTo, MoveTrolleyTo*) or relative (*RotateJibAngle, MoveTrolletDist*).
- Tool operate: to grasp (*Set*) and ungrasp (*Reset*) the modules.

An example of programme is the following:

```
MOVETO (1.90,5.90,1.60);
MOVETOZ (1.10);
SET ;
MOVETOZ (1.60);
MOVEJOINTS (0.90,2.10,1.60);
MOVETOZ (1.10);
RESET ;
MOVETOZ (1.60);
```

The crane goes to the supply point, then grasp the module. The module is raised and transported to the final location. The module is released and the crane moves back.

3.3 Program Editor

A program is a text file containing a sequence of commands. This file can be written directly in a text editor, can be generated automatically using a specific application or can be edited in a robot programming editor.
One of the planning tools (see figure 1) can generate automatically the program from the design data and the crane specifications. This application calculates the assembly sequence and the path that each module has to follow from the supply area to the final point avoiding collisions. This is the main source of written programs.

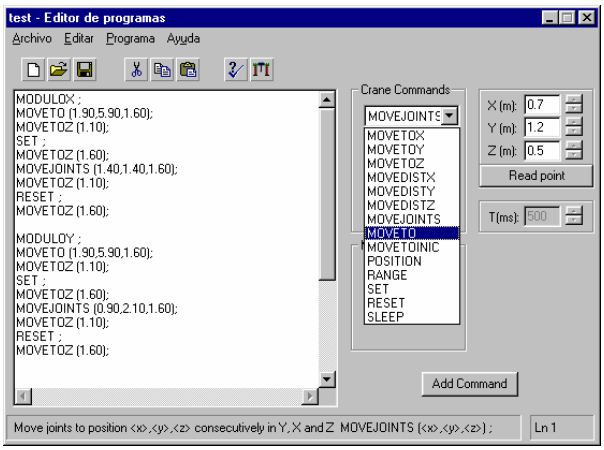

Figure 4. Program Editor

A program editor has been created to develop or modify programs. This editor is integrated in AutoCAD with the rest of applications and has an appearance similar to other windows applications (see figure 4). Commands are selected from a list and the editor automatically inserts the adequate parameters. The current position of the crane can be read directly from AutoCAD.

3.4 Graphical User Interface

All applications are integrated in AutoCAD, so AutoCAD is the main interface. Tools developed work like native AutoCAD commands, they are accessible from menus or command line.

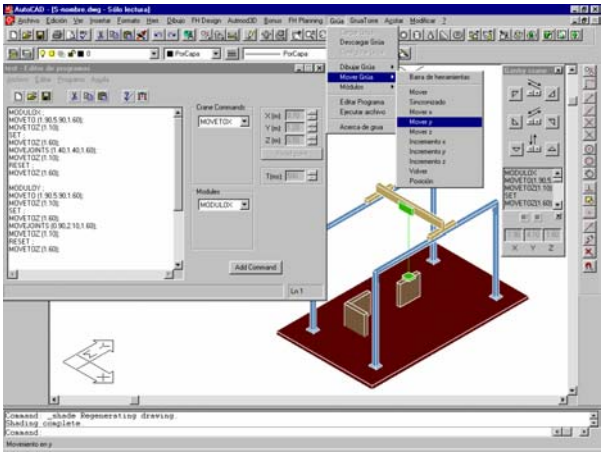
Figure 5. Graphical User Interface

The main features of the graphical interface (see figure 5) are the following:

- Specific menus integrated in AutoCAD which permit the applications use.
- Toolbar to launch program editor and run simulations.
- Program editor.

- Graphical pendant to handle every type of crane. This dialog box allows to move each joint individually, shows the current running instruction when a program is executing and shows the current position of the grasping platform.

4. SIMULATION EXAMPLES

To run a simulation the user selects this option from the menu, the toolbar or typing the instruction in the command line. Then the user selects the program previously created. There is two execution modes: instruction by instruction or continuous. Finally, the user select the speed of the simulation. To visualize the simulation, the application updates position and orientation of movable parts (crane joints and modules) at regular time intervals.

4.1 Gantry crane

Figure 6 shows four moments of a simulation sequence. The instructions list are written in the previous section. The assembly starts when a module arrives to the supply area (figure 6 upper left). Then the crane moves to this position and grasp the module (figure 6 upper right). The crane moves the module along the calculated path (figure 6 bottom left). Finally, the module is placed in its final position and the crane moves back (figure 6 bottom right).

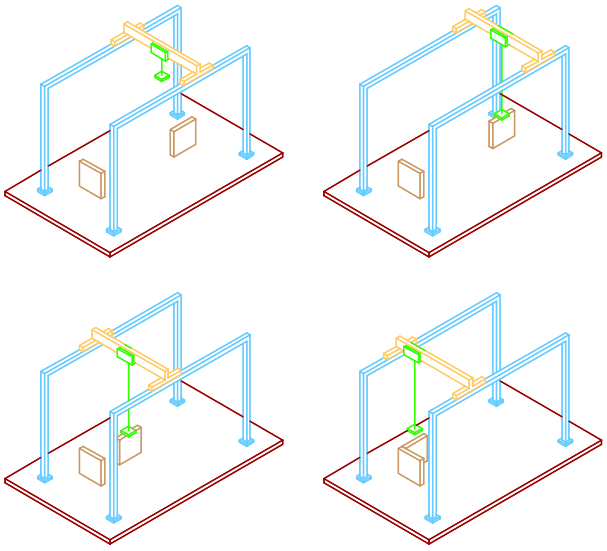

Figure 6. Simulation sequence with a gantry crane

4.2 Tower crane

Tower cranes need to rotate the module to place it in a correct orientation, because the slewing of the jib over the tower. To perform a similar task like that shown in the gantry crane case, the following program is needed:

```
MOVETO (6.0,0.0,4.0);
TOOLORIENT (0);
MOVETOZ (2.0);
SET ;
MOVETOZ (4.0);
MOVEJOINTS (0.80,7.30,4.0);
TOOLORIENT (0);
MOVETOZ (2.0);
RESET ;
MOVETOZ (4.0);
```

The difference is the instruction *ToolOrient* which permits the rotation of the module.
Figure 7 shows six moments of a simulation sequence. The assembly starts when a module arrives to the supply area (figure 7 upper left). Then the tower crane moves to this position and grasp the module (figure 7 upper right). The crane moves the module along the calculated path (figure 7 middle left) to the final position, but in a higher level (figure 7 middle right). Then the module descends (figure 7 bottom left) and finally it is placed in its final position and the crane moves back (figure 7 bottom right).

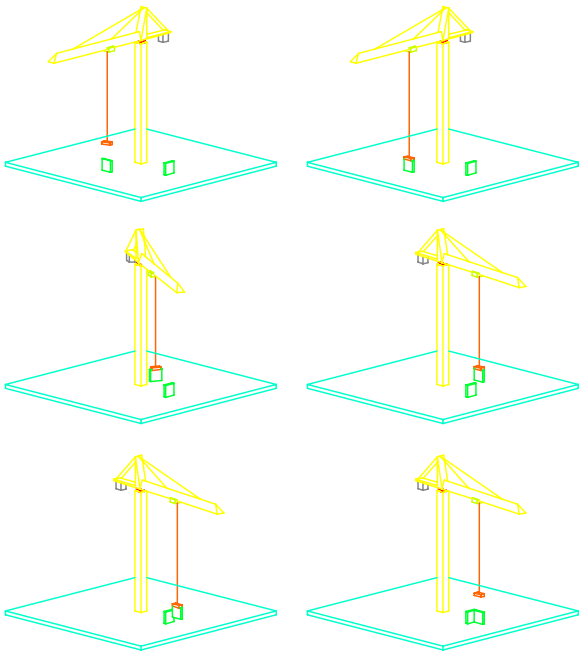

Figure 7. Simulation sequence with a tower crane

5. LABORATORY GANTRY CRANE

The automatic gantry crane developed at the Laboratory is used to test the automatic modules placement. The system was developed using a

commercial crane, which was modified by adding the adequate sensors, servomotors and an advanced control system [Abderrahim].

Figure 8. Assembly of a module with the laboratory gantry crane

Figure 8 shows four moments of an assembly sequence performed by the laboratory gantry crane. The grasping platform goes to the supply area and capture the module (figure 8 upper left). Then the crane rises the module to avoid collisions (figure 8 upper right) and moves it along the calculated path to the final position, but in a higher level (figure 8 bottom left). Finally, the module is placed in its final position and the crane moves back (figure 8 bottom right).

6. CONCLUSIONS

The developed crane simulators present new tools to test the automatic assembly of prefabricated building modules. The outcome demonstrates the feasibility of the process of automation and shows how the system could assemble 2D and 3D modules, where all movements are tested first on the simulator. The integration of several applications in the same graphical interface facilitates the information interchange and reduces time for learning and using. The system can be extended or updated since one of the main features is modularity.

7. ACKNOWLEDGEMENT

This work has been supported by the EU project FutureHome under BRITE program with reference nº ER4-29671 and in close coordination with the IF7 Japanese program in the frame of the IMS trans- regional program. The authors would like to thank people who have helped in the development of this project, especially to E. Navas, L. Camaño, and J. Castro.

8. REFERENCES

[Abderrahim] Abderrahim, M. Giménez, A. Nombela, A. Garrido, S. Diez, R. Padrón, V. M. and Balaguer, C. (2001). *The design and development of an automatic construction crane*. 18th International Symposium on Automation and Robotics in Construction (ISARC'2001), 149–154.

[Balaguer] Balaguer, C., Abderrahim, M, J.M. Navarro, S. Boudjaber, P. Aromma, K. Kahkonen, S. Slavenburg, D. Seward, R. Wing, B. Atkin, *FutureHome: An Integrated Construction Automation Approach*, IEEE Robotics and Automation magazine, 9(1), 55-66.

[Diez 2000] Diez, R. Abderrahim, M. Padrón, V. M. Celorrio, L. Pastor, J. M. and Balaguer C. (2000). *AUTMOD3: an automatic 3D modularization system*. 17th International Symposium on Automation and Robotics in Construction (ISARC'2000), 1033–1038.

[Diez 2003] Diez, R. (2003). *Automatización del diseño de edificios modulares para su construcción robotizada* (In Spanish). PhD Thesis, Carlos III University of Madrid.

[Lapham] Lapham, J. (1999). *RobotScript the introduction of a universal robot programming language*, Industrial Robot, 26(1), 17-25.

[McAuley] McAuley, C. (2000) *Programming AutoCAD 2000 Using ObjectARX*, Autodesk Press.

[Owen Ransen] Ransen, Owen. *AutoCAD programming in C/C++*, John Wiley & Sons, 1997.

[Padron] Padrón, V.M., Cárdenas, O., Diez, R., Abderrahim, M. and Balaguer, C. (2002). *AUTMOD3: A planning tool for modular building system*. 19th International Symposium on Automation and Robotics in Construction (ISARC'02), 91–96.